企业级 Web 集群服务器架构搭建项目文档

一、项目基础信息

（一）项目名称

企业级 Web 业务服务器集群架构搭建项目（含高可用、数据备份与自动化运维体系）

（二）项目背景与目标

1. 项目背景：为支撑多类型 Web 业务稳定运行，覆盖 PHP（问答社区、网盘）与 Java（博客）产品，解决单一服务器部署的单点故障、数据丢失、运维效率低等问题，搭建一套符合企业标准的高可用、可扩展集群架构。
2. 项目目标：
   - 业务支撑：兼容 Wecenter、Phpshe（PHP 环境）与 Zrlog（Java+Tomcat 环境）运行；
   - 高可用：实现 99.9% 服务可用性，支持负载均衡主备切换、数据库主从故障转移；
   - 数据安全：通过 rsync 实现全网备份与异地灾备，数据恢复时间 < 1 小时；
   - 运维高效：通过 Ansible 自动化管理、Zabbix 监控，降低集群维护成本；
   - 扩展性：支持 Web 节点、存储容量横向扩展，适配业务增长需求。

（三）项目范围

1. 核心服务：负载均衡、Web 服务集群、MySQL 主从数据库、NFS 共享存储、rsync 备份服务；
2. 支撑服务：NTP 时间同步、内网共享上网、防火墙防护、跳板机管理、Ansible 自动化、Zabbix 监控；
3. 不含范围：硬件采购、外网带宽申请、域名注册（仅含架构内域名解析配置）。

（四）文档信息

- 文档受众：课程老师、项目实施人员；
- 文档版本：V1.0（初稿）；
- 编写工具：Word（主体）、ProcessOn（架构图，待补充）。

二、需求分析

（一）功能需求

1. 业务支撑需求：
   - 部署 2 种 PHP 产品（ Wecenter/Phpshe）与 1 种 Java 产品（Zrlog）；

- PHP 产品运行环境：Nginx+PHP，Java 产品运行环境：Tomcat；
- 实现 HTTPS 加密访问，静态资源（图片 / 视频）集中存储。

2. 核心服务需求：
   - 负载均衡：2 台节点实现高可用，支持请求分发；
   - 数据库：主从复制、读写分离，保障数据一致性；
   - 备份：全网数据定时备份，支持异地灾备；
   - 存储：NFS 共享存储，实现 Web 节点附件同步。
3. 管理需求：
   - 内网服务器共享上网，公网节点配置防火墙；
   - 跳板机集中管理服务器，支持权限控制与审计；
   - 集群时间统一同步，自动化部署与配置。

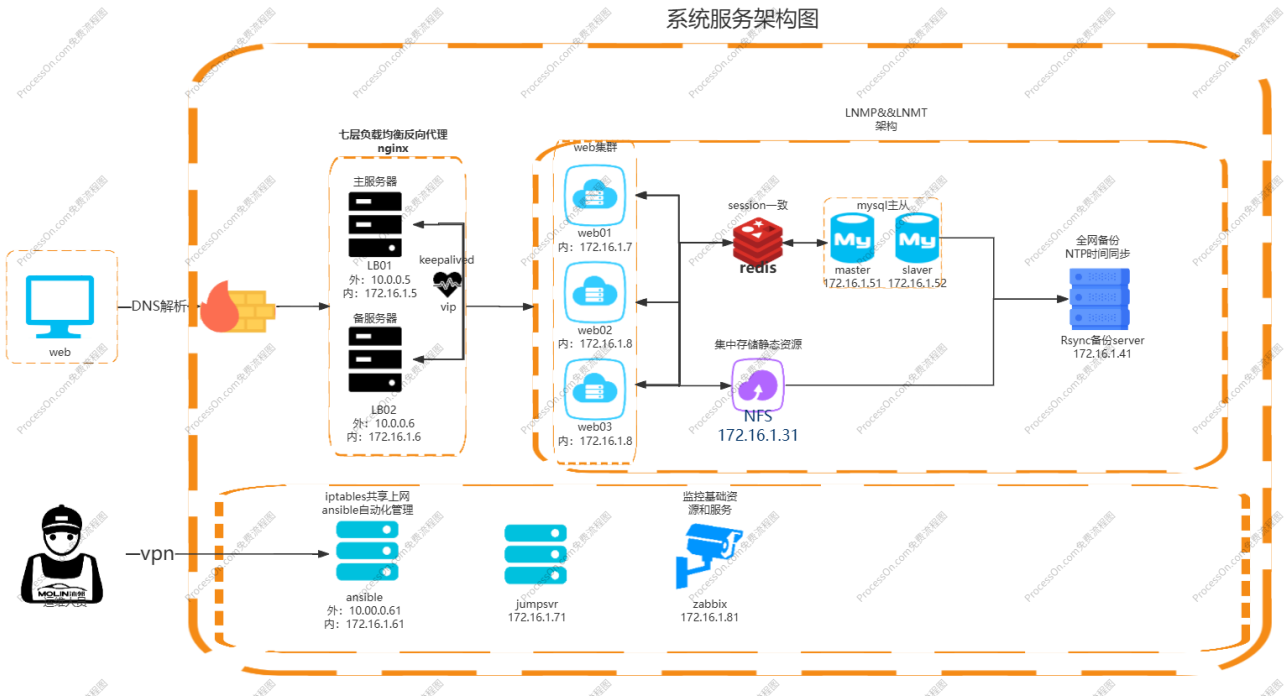
（二）非功能需求

1. 性能需求：Web 节点单台支持并发 1000+，数据库查询响应时间 < 500ms；
2. 可用性需求：负载均衡主备切换时间 < 10 秒，数据库主从故障转移 < 30 秒；
3. 安全性需求：公网节点仅开放 80/443/22 端口，敏感数据加密传输，符合等保合规；
4. 扩展性需求：支持新增 Web 节点、扩展存储容量，无需修改核心架构。

三、架构总体设计

（一）架构组成（架构图待补充）

1. 接入层：2 台负载均衡节点（Nginx+Keepalived），配置公网 IP 与内网 IP，实现高可用与请求分发；
2. 应用层：3 台 Web 服务器集群，部署 Nginx+PHP+Tomcat，挂载 NFS 共享存储，安装 rsync 客户端；
3. 数据层：1台Maxscale服务器，2台MySQL服务器构建主从架构（主库写、从库读），实现数据库读写分离；
4. 存储层：1 台 NFS 服务器，集中存储 Web 节点静态资源与附件；
5. 备份层：1 台 rsync 备份服务器，实现全网数据定时备份与异地灾备；
6. 管理层：2 台管理节点，部署 Ansible、跳板机、VPN，实现集群自动化管理与安全访问；
7. 监控层：1 台 Zabbix 服务器，监控所有节点基础资源与服务状态。



（二）核心设计思路

1. 高可用：关键服务（负载均衡、数据库）采用主备 / 主从架构，避免单点故障；
2. 性能优化：负载均衡分发请求，数据库读写分离，静态资源集中存储与 CDN 加速；
3. 数据安全：多层备份（本地备份 + rsync 全网备份 + 异地灾备），时间同步保障数据一致性；
4. 运维高效：自动化工具（Ansible）批量配置，监控系统（Zabbix）实时告警，跳板机统一管理。

（三）数据流向说明

用户请求 → 公网 → 负载均衡器（HTTPS 终止）→ Web 服务器集群 → 静态资源从 NFS 读取 / 动态请求转发至数据库代理（Maxscale）再至 MySQL（主库写 / 从库读）→ 全网数据定时同步至 rsync 备份服务器。

四、技术选型明细

角色	主机名	内网IP	公网IP	软件栈	用途
负载均衡01	lb01	172.16.1.5	10.0.0.5	Nginx + Keepalived	高可用请求分发，HTTPS 配置
负载均衡02	lb02	172.16.1.6	10.0.0.6	Nginx + Keepalived	高可用请求分发，HTTPS 配置
Web服务器01	web01	172.16.1.7	-	Nginx + PHP + Tomcat	运行 Web 业务，挂载 NFS 存储
Web服务器02	web02	172.16.1.8	-	Nginx + PHP + Tomcat	运行 Web 业务，挂载 NFS 存储

角色	主机名	内网IP	公网IP	软件栈	用途
Web服务器03	web03	172.16.1.9	-	Tomcat（Java环境）	运行 Web 业务，挂载 NFS 存储
数据库主	db01	172.16.1.51	-	MySQL	主从复制，读写分离
数据库从	db02	172.16.1.52	-	MySQL	主从复制，读写分离
存储服务器	nfs01	172.16.1.31	-	NFS	集中存储静态资源与附件
备份服务器	backup	172.16.1.41	-	Rsync + NTP	全网数据备份，NTP 时间同步
跳板机	jumpsrv	172.16.1.71	-	Jumpserver	跳板机
监控服务器	zabbix	172.16.1.81	-	Zabbix Server + Agent	集群基础资源与服务监控
自动化服务器	ansible	172.16.1.61	10.0.0.61	Ansible + VPN + 共享上网	自动化管理、跳板机、VPN 服务
数据库中间件服务器	maxscale	172.16.1.53		MaxScale	作为数据库代理服务器实现数据读写分离

选型说明

- 操作系统选择 CentOS 7.9，兼容性强、稳定性高，符合企业运维习惯；
- 核心软件选用稳定版本（如 Nginx 1.20.1、MySQL 8.0），兼顾性能与兼容性；
- 管理工具选用开源方案（Ansible、Jumpserver、Zabbix），降低成本且易维护。

五、详细实施步骤

（一）基础环境准备（所有节点通用）

1. 系统优化：
   - 关闭防火墙：`systemctl stop firewalld && systemctl disable firewalld`；
   - 禁用 Selinux：`sed -i 's/SELINUX=enforcing/SELINUX=disabled/' /etc/selinux/config`，重启生效；
   - 配置主机名：按节点类型命名（如 `hostnamectl set-hostname lb01`），修改 `/etc/hosts` 添加内网 IP 与主机名映射。
2. 目录规范：
   - 创建统一目录：`mkdir -p /server/tools /server/scripts /code /backup /data`；

- 站点目录划分：`mkdir -p /code/{wecenter,phpshe,zrlog}`。

3. 依赖安装：
   - 安装基础工具：`yum install -y wget vim net-tools gcc gcc-c++`。

（二）核心服务部署（按顺序实施）

1. NFS 共享存储节点部署（172.16.1.31）

1. 安装 NFS 服务：`yum install -y nfs-utils rpcbind`；
2. 配置共享目录：
   - 编辑配置文件：`vim /etc/exports`，添加 `/data 172.16.1.0/24(rw,sync,no_root_squash)`；
   - 创建共享目录：`mkdir -p /data && chmod 755 /data`；
3. 启动服务并设置自启：
   - `systemctl start rpcbind && systemctl enable rpcbind`；
   - `systemctl start nfs && systemctl enable nfs`；
4. 验证：`showmount -e 172.16.1.31`，显示共享目录即成功。

ansible步骤：
创建一个名为nfs_server的role，然后编写playbook来调用这个role。
具体步骤：

1. 创建role的目录结构
2. 编写role的任务（tasks）
3. 编写变量（vars，如果需要）
4. 编写handlers（用于重启服务）
5. 编写playbook来调用这个role

2. rsync 备份节点部署（172.16.1.41）

1. 安装 rsync：`yum install -y rsync`；
2. 配置 rsync 服务：
   - 编辑配置文件：`vim /etc/rsyncd.conf`，添加核心配置（uid=root、gid=root、备份目录 /backup、认证用户 backupuser、允许 172.16.1.0/24 网段访问）；
   - 创建备份目录(已存在)：`mkdir -p /backup && chmod 755 /backup`；
   - 配置密码文件：`echo "backupuser:123456" > /etc/rsyncd.secrets && chmod 600 /etc/rsyncd.secrets`；
3. 启动服务并设置自启：`systemctl start rsyncd && systemctl enable rsyncd`；
4. 验证：`netstat -tuln | grep 873`，显示 873 端口监听即成功；
5. 配置 NTP 时间同步：
   - 安装 ntp：`yum install -y ntp`；

- 编辑 `/etc/ntp.conf`，配置外网同步源（如ntp.aliyun.com），允许内网网段同步；
- 启动服务：`systemctl start ntpd && systemctl enable ntpd`。

3. MySQL 主从数据库部署

（1）主库部署（db01：172.16.1.51）

1. 安装 MySQL 8.0：通过 yum 源安装，配置 `/etc/my.cnf`（添加 server-id=1、log-bin=mysql-bin）；
2. 启动服务：`systemctl start mysqld && systemctl enable mysqld`；
3. 初始化配置：
   - 查看初始密码：`grep 'temporary password' /var/log/mysqld.log`；
   - 登录 MySQL，修改密码：`ALTER USER 'root'@'localhost' IDENTIFIED BY 'Root@123456';`；
   - 创建从库同步用户：`CREATE USER 'repl'@'172.16.1.52' IDENTIFIED BY 'Repl@123456';`，授权：`GRANT REPLICATION SLAVE ON *.* TO 'repl'@'172.16.1.52';`；
4. 锁定数据库并查看状态：`FLUSH TABLES WITH READ LOCK;`，`SHOW MASTER STATUS;`（记录 File 和 Position 值）。

（2）从库部署（db02：172.16.1.52）

1. 安装 MySQL 8.0，配置 `/etc/my.cnf`（添加 server-id=2、relay-log=relay-bin）；
2. 启动服务并初始化密码，登录后配置主从同步：
   - `CHANGE MASTER TO MASTER_HOST='172.16.1.51', MASTER_USER='repl', MASTER_PASSWORD='Repl@123456', MASTER_LOG_FILE='xxx', MASTER_LOG_POS=xxx;`（替换主库 File 和 Position 值）；
3. 启动同步并验证：`START SLAVE;`，`SHOW SLAVE STATUS\G`（确保 Slave_IO_Running 和 Slave_SQL_Running 均为 Yes）。

（3）数据库代理服务器部署（maxscale：172.16.1.53）

1. 安装maxscale，因为要适配centOS7，所以很难找到适合的maxscale的rpm包，但还是找到了。
2. 更改配置文件：
   在/etc/maxscale.cnf下添加：

```
[maxscale]
threads=4
log_info=1
syslog=1
```

```
datadir=/var/lib/maxscale
cachedir=/var/cache/maxscale
logdir=/var/log/maxscale
piddir=/var/run/maxscale

# 主库服务器定义
[server1]
type=server
address=172.16.1.51
port=3306
protocol=MariaDBBackend

# 从库服务器定义
[server2]
type=server
address=172.16.1.52
port=3306
protocol=MariaDBBackend

# 监听器配置
[RW-Split-Listener]
type=listener
service=RW-Split-Service
protocol=MariaDBClient
port=4006

# 读写分离服务
[RW-Split-Service]
type=service
"/etc/maxscale.cnf" 51L, 879C
[maxscale]
[hreads=4]
log_info=1
syslog=1
datadir=/var/lib/maxscale
cachedir=/var/cache/maxscale
logdir=/var/log/maxscale
piddir=/var/run/maxscale
# 主库服务器定义
[server1]
type=server
address=172.16.1.51
port=3306
protocol=MariaDBBackend
# 从库服务器定义
[server2]
type=server
```

```
address=172.16.1.52
port=3306
protocol=MariaDBBackend
# 监听器配置
[RW-Split-Listener]
type=listener
service=RW-Split-Service
protocol=MariaDBClient
port=4006
# 读写分离服务
[RW-Split-Service]
type=service
router=readwritesplit
```

3. 启动服务并验证：
   systemctl start maxscale
   journalctl -xe
   maxctrl list servers

4. Web 服务器节点部署

（1）web01/web02（PHP 环境部署）

1. 安装 Nginx+PHP：`yum install -y nginx php php-fpm php-mysql`；
2. 配置 Nginx：编辑 `/etc/nginx/conf.d/` 下站点配置（blog.conf、wecenter.conf），指向 `/code` 对应目录，配置 FastCGI 连接 PHP-FPM；
3. 挂载 NFS 存储：`mount -t nfs 172.16.1.31:/data /code/blog/uploads`（其他站点同理），添加至 `/etc/fstab` 实现开机自动挂载；
4. 安装 rsync 客户端：`yum install -y rsync`，配置定时任务：`0 3 * * * rsync -avz /code backupuser@172.16.1.41::backup --password-file=/etc/rsync.pass`；
5. 启动服务：`systemctl start nginx php-fpm && systemctl enable nginx php-fpm`；
6. 部署 PHP 产品：下载 Wecenter,Phpshe 源码至对应站点目录，配置数据库连接。

（2）web03（Java 环境部署）

1. 安装 JDK+Tomcat：解压 JDK 至 `/usr/local/jdk`，配置环境变量；解压 Tomcat 至 `/usr/local/tomcat`；
2. 部署 Zrlog：将 Zrlogwar 包放入 Tomcat 的 webapps 目录，启动 Tomcat：`/usr/local/tomcat/bin/startup.sh`；
3. 挂载 NFS 存储：`mount -t nfs 172.16.1.31:/data /usr/local/tomcat/webapps/zrlog/uploads`，添加至 `/etc/fstab`；

4. 配置 rsync 定时备份：同 web01/web02 配置。

5. 负载均衡节点部署（lb01/lb02）

1. 安装 Nginx+Keepalived：`yum install -y nginx keepalived`；
2. 配置 Nginx 负载均衡：编辑 `/etc/nginx/nginx.conf`，添加 upstream 指向 3 台 Web 节点，配置反向代理；
3. 配置 HTTPS：申请 SSL 证书，在 Nginx 配置中添加 ssl_certificate、ssl_certificate_key 等参数；
4. 配置 Keepalived：
   - lb01（主节点）：配置 `/etc/keepalived/keepalived.conf`，设置 state MASTER，virtual_ipaddress 为 10.0.0.3（VIP）；
   - lb02（备节点）：配置 state BACKUP，virtual_ipaddress 相同；
5. 启动服务：`systemctl start nginx keepalived && systemctl enable nginx keepalived`；
6. 验证：访问 VIP（10.0.0.3），可正常打开 Web 站点；关闭 lb01，VIP 自动漂移至 lb02。

（三）扩展服务部署

1. 跳板机部署（m02：172.16.1.71）

1. 安装 Jumpserver：执行一键安装脚本（参考官网：jumpserver.org）；
2. 初始化配置：访问http://172.16.1.71:8080，使用默认账号 admin/admin 登录，修改密码，添加服务器节点与运维用户；
3. 配置权限：按角色分配服务器访问权限，开启审计功能。

2. Ansible 自动化部署（m01：172.16.1.61）

1. 安装 Ansible：`yum install -y ansible`；
2. 配置主机清单：编辑 `/etc/ansible/hosts`，按节点类型分组（如 [web]、[db]、[lb]）；
3. 配置免密登录：`ssh-keygen` 生成密钥，`ssh-copy-id` 分发至所有节点；
4. 编写自动化剧本：创建初始化剧本（系统优化、依赖安装）、服务部署剧本（Nginx、MySQL），测试执行：`ansible-playbook init.yml`。

5. /etc/ansible/目录下的目录编排：

```
/etc/ansible/
├── ansible.cfg                 # Ansible 全局配置文件（可选，覆盖默认配置）
├── inventory/                  # 清单目录（按环境分类）
│   ├── production.ini          # 生产环境主机清单
│   ├── test.ini                # 测试环境主机清单
│   └── group_vars/             # 按主机组定义变量（全局）
│       ├── all.yml             # 所有主机通用变量（如通用路径、端口）
│       ├── webservers.yml      # web 组专属变量
│       └── dbservers.yml       # db 组专属变量
├── roles/                      # 核心：角色目录（按功能模块化）
│   ├── common/                 # 通用角色（如系统初始化、依赖安装）
│   │   ├── tasks/              # 任务列表（核心）
│   │   │   └── main.yml        # common 角色的主任务文件（必选）
│   │   ├── handlers/           # 处理器（如重启服务）
│   │   │   └── main.yml
│   │   ├── vars/               # 角色专属变量
│   │   │   └── main.yml
│   │   ├── templates/          # 模板文件（如配置文件模板）
│   │   └── files/              # 静态文件（如脚本、二进制包）
│   ├── nginx/                  # nginx 部署角色
│   │   ├── tasks/
│   │   │   └── main.yml
│   │   ├── handlers/
│   │   │   └── main.yml
│   │   ├── vars/
│   │   │   └── main.yml
│   │   ├── templates/
│   │   │   └── nginx.conf.j2
│   │   └── files/
│   ├── mysql/                  # mysql 部署角色
│   │   └── ...（结构同 nginx）
│   └── redis/                  # redis 部署角色
│       └── ...（结构同 nginx）
├── playbooks/                  # Playbook 目录（按模块/场景拆分）
│   ├── common.yml              # 仅执行 common 角色的 playbook
│   ├── nginx.yml               # 仅执行 nginx 角色的 playbook
│   ├── mysql.yml               # 仅执行 mysql 角色的 playbook
│   ├── redis.yml               # 仅执行 redis 角色的 playbook
│   └── full_deploy.yml         # 整合所有角色的总 playbook（一键部署）
└── scripts/                    # 辅助脚本（如 SSH 脚本、初始化脚本，可选）
    └── check_ssh.sh
```

3. Zabbix 监控部署（172.16.1.81）

1. 安装 Zabbix Server：通过 yum 源安装 Zabbix Server 与数据库（可复用 MySQL 主库）；
2. 配置 Zabbix Server：编辑 `/etc/zabbix/zabbix_server.conf`，指定数据库连接信息；
3. 安装 Zabbix Agent：所有节点安装 agent，配置 `/etc/zabbix/zabbix_agentd.conf`，指向 Zabbix Server；
4. 配置监控模板：添加主机，关联基础监控模板（CPU、内存、磁盘）与服务模板（Nginx、MySQL），设置告警规则。

六、测试与验证方案

（一）功能测试

1. Web 服务测试：访问 VIP（10.0.0.3），验证各站点（WordPress、Wecenter、Zrlog）正常访问、文章发布、图片上传；
2. 负载均衡测试：查看 Nginx 访问日志，确认请求均匀分发至 3 台 Web 节点；关闭 lb01，验证 VIP 漂移，服务不中断；
3. 数据库测试：主库创建数据，从库验证同步成功；模拟主库宕机，从库手动切换为主库，业务正常读写；

4. 备份测试：查看 rsync 备份节点 `/backup` 目录，确认每日 3 点自动备份成功；删除 Web 节点数据，通过备份文件恢复，验证数据完整性。

（二）性能测试

1. 并发测试：用 JMeter 模拟 1000 并发访问，Web 节点 CPU 使用率 < 70%，响应时间 < 500ms；
2. 数据库测试：模拟 1000 条并发查询，从库响应时间 < 300ms，主从同步延迟 < 1s。

（三）可用性测试

1. 单点故障测试：关闭单台 Web 节点，其余节点正常承接请求；关闭 MySQL 主库，从库切换后业务恢复；
2. 时间同步测试：所有节点执行 `ntpdate 172.16.1.41`，验证时间一致。

（四）测试结果记录

按测试类型整理测试用例、执行步骤、预期结果、实际结果，标注通过状态，形成测试报告。

七、风险预案与运维管理

（一）风险预案

可能风险	影响范围	应对方案
rsync 备份节点硬盘损坏	全网备份数据丢失	1. 备份节点配置 RAID5；2. 每周将 `/backup` 目录同步至异地存储（如对象存储）
MySQL 主库宕机	业务无法写数据	1. 配置 MHA 工具实现主从自动切换；2. 提前备份主库数据，紧急时手动恢复至从库
负载均衡 Nginx 配置错误	所有用户无法访问	1. 配置修改前备份 `nginx.conf`；2. 错误时执行 `nginx -t` 验证配置，恢复备份
NFS 存储节点故障	静态资源无法访问	1. Web 节点本地缓存静态资源；2. 部署备用 NFS 节点，配置实时同步

（二）运维管理

1. 定时任务：
   - 备份任务：rsync 每日 3 点自动备份，保留 30 天备份文件；
   - 日志切割：Nginx、MySQL 日志每日切割，删除 90 天前日志；
   - 巡检任务：每周一执行 Ansible 巡检剧本，检查服务状态与磁盘使用率。

2. 日志管理：所有节点核心日志（/var/log/nginx、/var/log/mysqld.log）集中收集至ELK（可选扩展），便于问题排查；
3. 巡检计划：
   - 每日：查看 Zabbix 告警，检查核心服务状态；
   - 每周：检查备份完整性、磁盘使用率（<80%）、数据库主从同步状态；
   - 每月：模拟单点故障，测试灾备恢复流程。

八、实施规范要求

（一）基础配置规范

1. 自启动要求：所有核心服务（Nginx、MySQL、rsync、keepalived）均设置开机自启；
2. 系统优化：关闭防火墙、禁用 Selinux，优化 Linux 内核参数（文件描述符、TCP 连接）；
3. 目录规范：严格遵循预设目录（/server/tools、/code、/backup、/data），禁止随意创建目录。

（二）命名规范

1. 主机名：按节点类型命名（lb01/lb02、web01/web02/web03、db01/db02）；
2. IP 地址：严格遵循规划（172.16.1.X 段），禁止随意修改；
3. 域名：oldboy.com解析至 VIP（10.0.0.3），各站点通过二级域名访问（如blog.oldboy.com）。

（三）项目周期规划

1. 总周期：8 天；
2. 阶段划分：
   - 第 1 天：完成项目文档与架构图绘制；
   - 第 2-3 天：搭建基础服务（NFS、rsync、MySQL 主从、Web 节点）；
   - 第 4-5 天：搭建负载均衡、HTTPS 配置、自动化工具（Ansible）；
   - 第 6-7 天：搭建扩展服务（跳板机、VPN、Zabbix 监控）；
   - 第 8 天：测试验证、问题修复、文档整理。

九、附录

（一）关键术语解释

- NTP：网络时间协议，用于集群节点时间同步；
- NFS：网络文件系统，实现多节点共享存储；
- rsync：文件同步工具，支持增量备份，用于全网数据备份；

- Ansible：自动化运维工具，实现批量配置与部署；
- Zabbix：开源监控系统，监控节点资源与服务状态；
- Keepalived：高可用工具，实现负载均衡节点 VIP 漂移。

（二）参考资料

- Nginx 官方文档：https://nginx.org/en/docs/；
- MySQL 主从复制文档：https://dev.mysql.com/doc/refman/8.0/en/replication.html；
- Ansible 官方指南：https://docs.ansible.com/；
- Zabbix 官方文档：https://www.zabbix.com/documentation；
- Jumpserver 安装指南：https://jumpserver.org/docs/。

（三）联系方式

- 项目负责人：汤嘉城；
- 技术支持：刘海（课程老师）。